ISSN: 1692-7257 - Volumen 2 – Número 30 - 2017

Revista Colombiana de Tecnologías de Avanzada

https://doi.org/10.24054/16927257.v30.n30.2017.2748

Recibido: 14 de enero de 2017
Aceptado: 22 de mayo de 2017

UNA HERRAMIENTA PARA LA ELABORACIÓN DE INTERFACES GRÁFICAS PARA SISTEMAS DE MONITOREO Y CONTROL USANDO ARQUITECTURA ARM Y .NET MICRO FRAMEWORK.

A TOOL FOR GRAPHICAL INTERFACES DESIGN TO USE IN MONITORING AND CONTROL SYSTEMS USING ARM ARCHITECTURE Y .NET MICRO FRAMEWORK

MSc. Carlos Mario Correa Torres, Tec. Jhon Fredy Valencia Gómez

Servicio Nacional de Aprendizaje, SENA.
TICs y Electrónica, Semillero de investigación en sistemas embebidos SISEMB.
Carrera 68 #104 Complejo Norte, Medellín, Antioquia, Colombia.
Tele.: (4) 4442800, E-mail: {cmcorrea, jfvalenciag}@sena.edu.co.

Resumen: El propósito de este trabajo es presentar herramientas de hardware y software para elaborar aplicaciones con sistemas embebidos que permiten desarrollos electrónicos con alta funcionalidad y desempeño; Microcontroladores de 32 bits con arquitectura *Advanced Risc Machine* (ARM) y entornos de desarrollo integrado (IDE) como *.Net Micro framework* que utiliza programación orientada a objetos. Basado en estas tecnologías, este artículo presenta una herramienta de software desarrollada en Visual Studio que permite diseñar interfaces gráficas de forma rápida y simple para implementar sistemas de control y monitoreo que pueden ser utilizados en entornos académicos e industriales.

Palabras clave: Arquitectura de procesadores, Entornos de Desarrollo Integrado, Interfaces Gráficas, Sistemas Embebidos.

Abstract: The purpose of this paper is to show hardware and software tools to develop applications with embedded systems that allow electronics developments with high functionality and performance; 32-bit microcontrollers with Advanced Risc Machine (ARM) architecture and integrated development environments (IDE) such as .Net Micro framework, that use object-oriented programming. Based on these technologies, this article presents a software tool developed in Visual Studio that allows designing graphical interfaces quickly and simply, to implement control and monitoring systems to be used in academic and industrial environments.

Keywords: Processor architecture, Integrated Development Environment, Graphical Interfaces, Embedded Systems.

1. INTRODUCCIÓN

La evolución de las arquitecturas de hardware para sistemas embebidos ha permitido que estos puedan ahora utilizar mayores recursos de memoria y aumentar el procesamiento de instrucciones por unidad de tiempo, siendo capaces de realizar funciones cada vez más diversas y complejas manteniendo la característica de un bajo consumo de potencia. Los procesadores con arquitectura *Advanced Risc Machine* (ARM) han conquistado el mercado de estos dispositivos permitiendo estandarización en algunos aspectos y logrando que mucho del trabajo se simplifique, que el tiempo de

desarrollo de productos electrónicos sea menor y se realice a un menor costo. Pueden encontrarse aplicaciones de su uso en la industria biomédica (Sandeep Raj, *et al.*,2015), (Humberto Loaiza, *et al*.,2014), en la industria automotriz (Kristian Ismail *et al*.,2015), en el desarrollo de hardware (Petrvalsky y Drutarovsky, 2016) entre otros.

La programación orientada a objetos hace ahora también parte de las nuevas herramientas de software para programación de sistemas embebidos, códigos administrados o interpretados son ventajas que pueden aplicarse en sistemas microcontrolados gracias a las nuevas características de hardware que estos poseen. Aplicaciones de estas plataformas de software tales como .*Net micro framework* pueden encontrarse en (Carlson *et al*., 2013) y (Babiuch, 2014). Este trabajo pretende evidenciar que con el uso estas nuevas herramientas pueden desarrollarse productos electrónicos con alta funcionalidad y desempeño, permitiéndonos ser más competitivos tanto en el desarrollo electrónico como en el desarrollo de herramientas que promuevan una educación pertinente en competencias, planteando escenarios reales de ejecución como se sugiere en (Giraldo *et al*, 2016). Se presenta en este artículo una herramienta de software diseñada en visual Studio que permite elaborar interfaces gráficas de forma rápida y simple para ser usada en sistemas de monitoreo control electrónico tanto para entornos académicos e industriales.

2. ARQUITECTURA ARM

El uso de microcontroladores de 4, 8 y 16 bits masificado en décadas anteriores ha venido disminuyendo mientras que en los últimos años microcontroladores de 32 bits se han posicionado como líder del mercado. El crecimiento en la participación del mercado de estos dispositivos y su proyección se evidencia en (MCclean, 2016) Fig1. Estas tendencias muestran la necesidad de su estudio y por tanto del estudio de arquitectura líder en ellos, la arquitectura ARM.

Creada inicialmente por la compañia Arcorn a mediados de la década de los 80, la arquitectura ARM es un tipo de arquitectura *Reduced Instruction Set Computer (*RISC) de 32 bits que busca minimizar el uso de recursos a partir de hardware de alto desempeño. ARM Holding no es en sí fabricante de silicio, solo licencia sus diseños lo que permite a diferentes fabricantes utilizar sus patentes para producir sus propios procesadores.

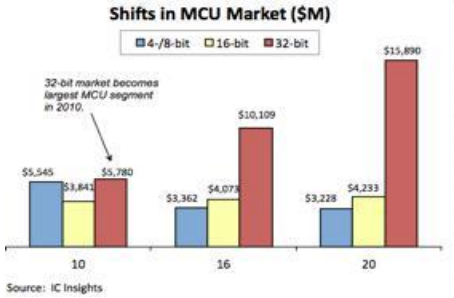

Fig. 1. Informe McClean 2016 Mercado de microcontroladores de 8, 16 y 32 Bits. (Fuente IC Insights).

Entre sus principales características se encuentran conjuntos de instrucciones ARM (32 bits) y Thumb (16 bits), el emplear códigos densos para lograr menos accesos a memoria, instrucciones que se compilan con códigos de condición, ALU que permite un desplazamiento lógico o aritmético sin consumo de tiempo, flujo de ejecuciones igual para casi todas las instrucciones y por lo general osciladores que trabajan a frecuencias no muy altas logrando así un reducido consumo de energía. Las últimas versiones de estas arquitecturas, constituyen familias organizadas de acuerdo a los requerimientos. Las familias Cortex A-R-M en la versión 7 se especializan en sectores específicos así (ARM, 2017):

- Familia ARM Cortex-A: Aplicaciones para sistemas Operativos y de aplicaciones específicas de alto desempeño.
- Familia ARM Cortex-R: Procesadores Embebidos para sistemas en tiempo Real, procesamiento de señales, aplicaciones de control crítico.
- Familia ARM Cortex-M: Procesos orientados a sistemas embebidos, Microcontroladores y Aplicaciones *System On Chip (*SoC).

La arquitectura ARM ha permitido una mayor estandarización para el control de hardware a través del *Cortex Microcontroller Software Interface Standard* (CMSIS) que es la herramienta de software que permite la interfase del procesador y sus periféricos, además de la configuración de los sistemas operativos de tiempo real (RTOS) y el *middleware*, esta es independiente del fabricante del chip y es transversal a las familias Cortex M, este aspecto permite mudar de plataforma sin mucha dificultad y consigue que la integración con otros sistemas resulte más sencilla, solucionando en parte el problema de estandarización presentado en (Ruben Sanchez, 2013).

En el desarrollo de la herramienta para la elaboración de interfaces gráficas se utilizó la placa de desarrollo STM32F429I-DISC0, esta placa se muestra en la Figura 2, y posee como características principales un microcontrolador ST Cortex M4 con oscilador 180 MHZ, 256 Kbytes de memoria RAM, 2MB de memoria flash y una pantalla de cristal liquido TFT táctil de 2.4 pulgadas. Las características más detalladas pueden encontrarse en (ST Microelectronics, 2016).

Fig. 2. Placa de desarrollo stm32f429I Disc0

3. LA PLATAFORMA .NET Y .NET MICROFRAMEWORK

Usando el modelo de programación orientado a objetos .NET es una plataforma de código administrado que permite el desarrollo de software usando diferentes lenguajes como Visual Basic .NET, C#, C++, entre otros. Desarrollada por Microsoft y orientada básicamente a aplicaciones de escritorio, está compuesta principalmente por un Entorno de Ejecución *Common Language Runtime* (CLR) que permite hacer la intermediación entre los diversos lenguajes y las capas más bajas del Hardware, además de los compiladores para los diferentes lenguajes .NET contiene amplias bibliotecas de Funciones, Herramientas de desarrollo, documentación y una gran comunidad de desarrolladores que comparten códigos. *.NET Micro framework* es a su vez una plataforma de software gratuito orientada al desarrollo de aplicaciones para sistemas embebidos que hereda muchas de las características de .NET, compuesto por un entorno de ejecución Tiny CLR (una versión ligera del CLR), *NET Micro Framework* es una extensión que puede instalarse en Visual Studio para gestionar código desarrollado en C# ,este utiliza un subconjunto de las bibliotecas .NET y permite una gestión más transparente para la configuración y gestión de hardware y periféricos tales como los GPIO, ADC, UART, SPI, *Timers*, entre otros. Es requisito de Hardware para poder ejecutarse que los sistemas embebidos que se programen posean como mínimo 256 Kbytes de memoria flash y 64 Kbytes de RAM, Características que satisfacen los microcontroladores ARM Cortex M3 y superiores. Una información más detallada puede encontrarse en (netmf.com, 2017).

4. HERRAMIENTA DESARROLLADA PARA EL DISEÑO PARA INTERFACES

El trabajo con una nueva herramienta de hardware en sistemas embebidos requiere de un estudio, algunas veces complejo, de su arquitectura y las bibliotecas de funciones que en muchas ocasiones el fabricante suministra en compiladores propios o que recomienda. Respecto al IDE para el desarrollo de programas, aunque la plataforma *.Net Micro framework* en conjunto con el CMSIS facilita el trabajo incorporando las bibliotecas de los principales módulos y periféricos, el trabajo gráfico resulta un poco dispendioso, es por eso que desarrollar una herramienta resulta útil, esta herramienta integrada a *.Net micro framework* permite construir y configurar objetos gráficos para la programación de sistemas de control con interfaces gráficas de forma simple.

El software desarrollado se compone principalmente de elementos comunes, indicadores y controles que a través de una selección y arrastre pueden ser ubicados en el área de la pantalla para el diseño de la interfaz, una vez se compila la solución la herramienta genera el código en C# de los objetos gráficos seleccionados y este es asociado a un proyecto donde el usuario podrá interactuar con cada uno de ellos para darle funcionalidad específica. El software incluye también pestañas para la configuración del Hardware. En la Figura 3 se presenta el entorno de la herramienta.

Fig. 3. Entorno de la Herramienta desarrollada para el diseño de interfaces gráficas

A continuación se describe brevemente la funcionalidad de los elementos de los que se compone el software:

- ***Text***: Escribe caracteres alfanuméricos ya sea de manera estática o dinámica durante la ejecución del programa.
- ***Chek Box***: Permite realizar selección de un listado de ítems.
- ***Image***: Carga y ubica imágenes en la pantalla.
- ***Button***: Dibuja un control tipo botón para que al presionar realice una función determinada por el programador.
- ***List Box***: Permite hacer listados para selección.
- ***Gauge***: Visualiza el valor de una variable en un instrumento indicador de aguja, permite escalizar en la función inicial.
- ***Progress Bar***: Visualiza barras de progreso para indicar progreso o nivel a partir de una variable, puede configurarse color, tamaño, orientación y permite escalizar en la función inicial.
- ***Motor***: Visualiza la activación y desactivación del eje de un motor y su dirección de giro puede configurarse a través de programación.
- ***Indicator***: Grafica indicadores lumínicos tipo LED, permite configurar color encendido, apagado y tamaño
- ***Plot***: Grafica estados de variables de forma continua en un plano X-Y, permite escalización en la función inicial.
- ***Keypad***: Permite usar un teclado alfa-numérico.
- ***Timer***: Presenta un temporizador que puede ser configurable por software o a través de la pantalla táctil una vez creado.
- ***Numeric***: Genera un control numérico que puede configurarse a través de software o a través del táctil.

Cada control contiene campos para configurar sus características propias, y estos controles pueden ser usados múltiples veces en un proyecto. La Herramienta posee para configuración de Hardware:

- ***PINS I/O***: Configura pines de entrada y salida para señales ON_OFF.
- ***ADC***: Configura el módulo del conversor analógico digital paras entradas analógicas.
- ***DAC***: Configura el módulo del conversor digital analógico paras salidas analógicas.
- ***PWM***: Configura salidas moduladas por ancho de pulso (PWM).
- ***Serial Port***: Configura el módulo para comunicaciones seriales UART.
- ***SPI***: Configura el módulo para comunicaciones seriales tipo SPI.
- ***Thread***: Configurar múltiples hilos de ejecución.

La Herramienta permite crear aplicaciones que incluyan múltiples ventanas en un mismo proyecto, presentando una lista de los elementos con sus respectivos índices y número de ventana donde fueron usados, permite además guardar los proyectos para ser posteriormente modificados**.**

5. EJEMPLOS DE APLICACIÓN DE LA HERRAMIENTA

A continuación se muestran algunas de las interfaces desarrolladas como ejemplos para prácticas con diferentes aplicaciones.

5.1 Implementación de un sistema de control de motores temporizado

El objetivo es diseñar y programar una interfaz gráfica que automatice el funcionamiento secuencial de dos motores que se activan por tiempos preestablecidos, estos tiempos pueden configurarse a través de dos controles tipo ***timer*** usando la pantalla táctil, se incluyen controles tipo ***motor*** para visualizar el estado de los motores y controles tipo ***image*** para cargar las imágenes de estos, se contabiliza además el número de ciclos usando un control tipo ***text***, también se activan salidas de control para que puedan ser conectadas físicamente.

La implementación de la interfaz en la tarjeta se muestra en la Fig 5.

Fig.5. Control de 2 motores temporizado implementado en la tarjeta STM32f429IDisc0.

5.2 Implementación de una interfaz gráfica para control de presión en caldera

En esta práctica se busca la configuración y visualización de variables analógicas. El sistema monitorea la presión en una caldera y para esto se diseña una interfaz que incluye un control tipo ***Gauge*** para la visualización de esta variable, un control tipo ***text*** para la indicación numérica del valor actual de la variable, tres controles ***indicator*** para niveles predeterminados de Presión baja, media, alta, dos controles tipo ***button*** para activar salidas correspondientes a una válvula y una bomba, además un control tipo ***plot*** para el registro continuo en el tiempo de la variable. La implementación de la interfaz en la tarjeta se muestra en la Figura 6.

Fig. 6. Interfaz de control de presión. Implementado en la tarjeta STM32f429IDisc0.

5.3 Implementación de Interfaz gráfica para el control del servomotor

En esta práctica se busca realizar una secuencia de 4 posiciones angulares de un servomotor, la interfaz permite configurar dichas posiciones en valores entre 0 y 180 grados que se configuran a través del uso del teclado de la herramienta, esta interfaz incluye 6 controles tipo ***image*** correspondientes al teclado pequeño que conducen a una segunda ventana donde se encuentra el teclado alfanumérico ***Keypad*** para configurar cada posición del servomotor digitando el valor en grados que se desea, también permite configurar el número de repeticiones que se desea realizar la secuencia. La interfaz incluye 7 indicadores tipo ***text*** , uno por cada posición configurada, uno que muestra la posición actual del servomotor (**C. POS**), otro para indicar el número de repeticiones configuradas (**P.C**)y el último para visualizar el ciclo en el que se encuentra (**C.C**). La interfaz se visualiza en las figuras 7 y 8.

Fig. 7. Pantalla principal Interfaz de control de posición con servo motor. Implementado en la tarjeta STM32f429IDisc0

Fig. 8 Pantalla auxiliar con teclado para configuración de las posiciones

5.4 Implementación de una interfaz gráfica para control de nivel

Esta práctica pretender a través de la pantalla táctil monitorear y controlar el nivel en un tanque. Posee controles tipo ***image*** que se usaron para cargar las imágenes correspondientes al tanque y las bombas de llenado y vaciado , controles tipo ***button*** para activar o desactivar los actuadores y para cambiar de ventana , la ventana 2 es una ventana solo para visualización que incluye controles tipo ***indicator*** que se activan en nivel bajo y alto y un control tipo ***plot*** para monitorear la variable de forma continua, incluye también esta ventana un indicador numérico a través de un control tipo ***text*** y un control tipo ***gauge*** que muestran el valor del nivel actual. La ventana principal y la segunda ventana de esta implementación se muestran en las figuras 9 y 10 respectivamente.

Fig. 9 Ventana 1 Interfaz de control de Nivel Implementado en la tarjeta STM32f429IDisc0

Fig. 10. Ventana 2 Interfaz de control de Nivel Implementado en la tarjeta STM32f429IDisc0

6. CONCLUSIONES

Es evidente que los dispositivos electrónicos demandan cada vez mayores requerimientos en funcionalidad y desempeño, para satisfacer estos requerimientos y ser competitivos en el desarrollo electrónico es necesario apoyarse en las nuevas herramientas disponibles de software y hardware que además permiten que estos desarrollos se realicen en menor tiempo y a un menor costo. La plataforma *.Net micro framework* a través de lenguaje C# facilita considerablemente la programación de sistemas embebidos y el manejo de periféricos para algunas arquitecturas ARM, este lenguaje, común para desarrolladores de aplicaciones de escritorio, acerca a estos al trabajo con hardware embebido y permite a todos aprovechar las ventajas que trae consigo el uso de un lenguaje de más alto nivel. Debe considerarse que *.Net Micro framework* no puede implementarse en cualquier sistema embebido, es por esto, que en trabajos futuros, se buscará elaborar una herramienta de uso más general, esto se logrará desarrollándola con base en bibliotecas escritas en lenguajes más básicos como lo son por ejemplo el C ó C++.

REFERENCIAS

Sandeep Raj, Chand G.S.S. Praveen, Chandra Kailash. (2015). *ARM-based arrhythmia beat monitoring system*, Microprocessors and Microsystems, 39, 504-511.

Loaiza Humberto, Ferrin. Carlos, Magdalena Ximena, López Steven, Henao Sebastián. (2014). *Sistema de extracción automática de parámetros morfológicos de la huella plantar mediante técnicas de visión por computador en un sistema embebido*. Revista Colombiana de Tecnologías de Avanzada. Revista Colombiana de Tecnologías de Avanzada, 23, 80-86.

Ismail Kristian, Muharam Aam, Pratama Mulia. (2015). *Design of CAN Bus for Research Applications Purpose Hybrid Electric Vehicle Using ARM Microcontroller*, Energy Procedia, 68, 288-296.

Petrvalsky Martin, Drutarovsky Milos. (2016). *Constant-weight coding based software implementation of DPA countermeasure in embedded microcontroller*, Microprocessors and Microsystems, 47, 82-89.

Carlson Jay, Mittek Mateusz, Pérez L. (2004). *Exploring the microsoft .NET micro framework for prototyping applied Wireless*

Sensor Networks, Electro/Information Technology (EIT), 1-6.

Babiuch Marek. (2014). .*Net Micro Framework gadgeteer measurement applications development*, Control Conference (ICCC), 15th International Carpathian, 10-13.

Wyrwol Bernard, Hrynkiewicz Edward. (2015) *Implementation of the FITA Fuzzy Inference System on the specific microcontroller platform*, FAC-Papers On Line, 48, 165-169.

Castillo Alejandro, Vázquez Javier, Ortegón Jaime Carrasco Roberto, Sandoval Jesús, Colli-Menchi Adrian. (2015). *A high-accuracy photovoltaic emulator system using ARM processors*, Solar Energy, 120, 389-398.

Giraldo Jorge, Ruiz Maryem, Rosero Claudia Zapata Luis. (2016). *Formación en competencias específicas para la industria del software colombiano. Experiencias del uso del aprendizaje basado en proyectos*. Revista Colombiana de Tecnologías de Avanzada, 27, 7-13.

Sánchez Rubén. (2013). *Estado del arte del desarrollo de sistemas embebidos desde una perspectiva integrada entre el hardware y software*. Revista Colombiana de Tecnologías de Avanzada, 22, 80-86.

INFORME McCLEAN. (2016). Mercado de microcontroladores de 8, 16 y 32 bits. http://www.icinsights.com/services/mcclean-report/report-contents

ARM. (2017). Clasificación de las familias ARM https://www.arm.com/products/processors.

STMICROELECTRONICS. (2016). Características de la Tarjeta de Desarrollo STM32f429IDisc0 http://www.st.com/en/evaluation-tools/32f429idiscovery.html.

.NETMF.(2016). https://www.microsoft.com/en-us/download/details.aspx?id=8515.